Functional Safety for Programmable Electronics Used in PPE: Best Practice Recommendations (In Nine Parts)

Part 3 - Functional Safety by Design

Prepared by Safety Requirements, Inc.
NIOSH Contract 200-2003-02355,
September 2007

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

FOREWORD

Background

Manufacturers of PPE use electronics and software technology to improve the safety of emergency responders and increase the likelihood of survival of victims. Electronics and software components embedded in PPE now provide protection, monitoring, and communication functions for emergency responders.

For example, innovative electronics and software engineers are accepting the challenge to design PPE that reduce reliance on audible communications. These products use radio and cellular frequencies to communicate digital information to the unit commander and among the various emergency responder agencies present on scene (i.e. police, fire, and rescue).

Innovators are also embedding electronics in turnout gear and taking advantage of newer materials. The result is more complex products including those that integrate products developed by different manufacturers. Although use of electronics and software provides benefits, the added complexity, if not properly considered, may adversely affect worker safety.

The Report Series

The report series contains best practice recommendations for the design and implementation of personal protection equipment and systems (PPE). The best practice recommendations apply to systems, protection layers, and devices using electronics and software embedded in or associated with PPE. The entire series provides information for use by life safety equipment manufacturers including component manufacturers, subassembly manufacturers, final equipment manufacturers, systems integrators, installers, and life safety professionals.

The reports in this series are printed as nine individual circulars. Figure 1depicts all nine titles in the series.

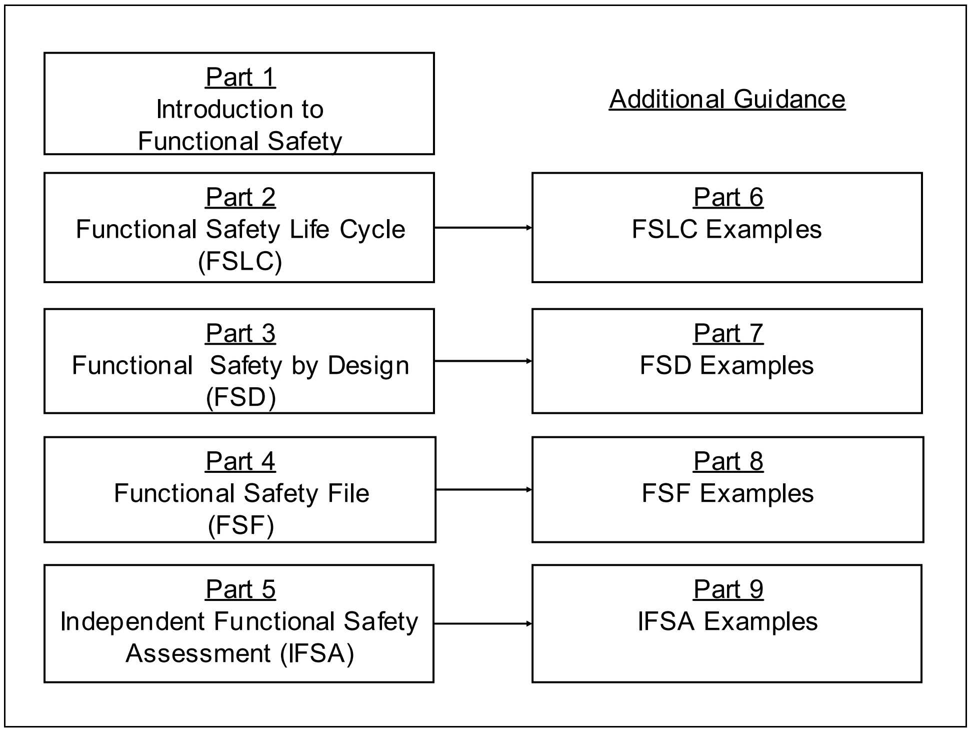
Part 1
Introduction to Functional Safety
Additional Guidance
Part 2
Functional Safety Life Cycle (FSLC)
Part 6
FSLC Examples
Part 3
Functional Safety by Design (FSD)
Part 7
FSD Examples
Part 4
Functional Safety File (FSF)
Part 8
FSF Examples
Part 5
Independent Functional Safety Assessment (IFSA)
Part 9
IFSA Examples

Figure 1 - The functional safety report series.

Report Scopes

Part 1: Introduction to Functional Safety

Part 1 is intended as an introductory report for the general protective equipment industry. The report provides an overview of functional safety concepts for advanced personal protective equipment and discusses the need to address them. The report also describes the practical benefits of implementing functional safety practices.

Part 2: The Functional Safety Life Cycle (FSLC)

Part 2 of the guidance recommends criteria for a Functional Safety Life Cycle. The use of a functional safety life cycle assures the consideration of safety during all phases of developing personal protection equipment and systems (PPE) from conceptualization to retirement, thus reducing the potential for hazards and injuries. The FSLC adds additional functional safety design activities to the equipment life cycle. FSD activities include identifying hazards due to functional failures, analyzing the risks of relying on electronics and software to provide functions, designing to eliminate or reduce hazards, and using this approach over the entire equipment life cycle. These activities start at the equipment level and flow down to the assemblies, subsystems, and components.

Part 3: Functional Safety by Design (FSD)

Functional safety seeks to design safety into the equipment for all phases of its use. Electronics and software are components; therefore, design of these components must take into account the overall achievement of functional safety. Part 3, Functional Safety by Design (FSD) provides best practice design criteria for use by manufacturers of PPE. The Mining industry guidelines prepared by NIOSH, MSHA and the mining industry manufacturers and entitled Programmable Electronic Mining Systems: Best Practices Recommendations (in Nine Parts)[1] serves as a basis for these guidelines. The report also draws from the design criteria found in International Electro-technical Commission (IEC) Standard 61508 Functional Safety of E/EE/PE Safety Related Systems[2] and the American National Standards Institute(ANSI) by Underwriters Laboratories(UL) 1998 Standard for Safety – Software in Programmable Components[3].

Part 4: Functional Safety File (FSF)

Part 4, Functional Safety File (FSF), details best practices for safety documentation through the development of a document repository named the FSF. Capturing safety information in the FSF repository starts at the beginning of the FSLC and continues during the full life cycle of the system. The FSF provides the documented evidence of following FSLC and FSD guidance in the report series. In essence, it is a “proof of safety” that the system and its operation meet the appropriate safety requirements for the intended application.

1 NIOSH Mining Industry Circulars 9456, 9458, 9460, 9461, 9464, 9487, 9488 Programmable Electronic Mining Systems: Best Practices Recommendations, 2001-2002. For further detail, see http://www.cdc.gov/niosh/mining/pubs. Date accessed: October 31, 2006.

2 IEC 61508 Functional Safety of E/EE/PE Safety Related Systems. For further detail, see http://www.iec.ch/61508 . Date accessed October 31, 2006

3 ANSI UL 1998 Standard for Safety: Software in Programmable Components. For further detail, see http://www.ul.com/software/ansi.html . Date accessed October 31, 2006.

Part 5: Independent Functional Safety Assessment (IFSA)

Part 5, Independent Functional Safety Assessment (IFSA), describes the scope, contents, and frequency of conducting IFSAs. The IFSA is an assessment of the documented evidence of the FSLC activities and FSD practices.

Part 6, 7, 8 and 9: Functional Safety - Additional Guidance

The Additional Guidance Reports consists of Parts 6, 7, 8, and 9 of the report series, and provides additional detail, which will help users to apply the functional safety framework.

The Parts 6, 7, 8 and 9 guidance information reinforces the concepts, describes various methods and tools that can be used, and gives examples and references. The guidance reports are not intended to promote a single methodology or to be an exhaustive treatise of the subject material. They provide examples and references so that the user may intelligently choose and implement the appropriate approaches given the user's application as follows:

- Part 6 – Additional Guidance: Functional Safety Life Cycle Examples are used to develop the Scope of the Project Plan. The scope guides Project Functional Safety by Design (FSD) Compliance and Project Documentation.
- Part 7 – Additional Guidance: Functional Safety by Design Examples drives Project Design for Safety Compliance, which then becomes part of the Project Documentation.
- Part 8 – Additional Guidance: Functional Safety File Examples help to complete the Project Documentation, to enable a third party assessment.
- Part 9 – Additional Guidance: Independent Functional Safety Audit Examples are employed in the development of the Third Party Assessment Report. Figure 2 overviews the relationships among Parts 6, 7, 8, and 9.

Part 6– Additional Guidance: Functional Safety Life Cycle (FSLC) Examples

Many manufacturers are ISO 9001 compliant as a result of requirements in NFPA codes and standards, follow Six Sigma approaches, and are using the Department of Defense (DoD) Software Engineering Institute (SEI) Capability Maturity Model (CMM) to improve life cycle practices. Part 6 provides a re-usable baseline FSLC Project Management Template (FSLC-PMT) that integrates these approaches. It also introduces the case example of DKYS, Device that Keeps You Safe to illustrate an FSLC. Appendix A of Part 6 is a general review of project management tools available to manage the FSLC activities.

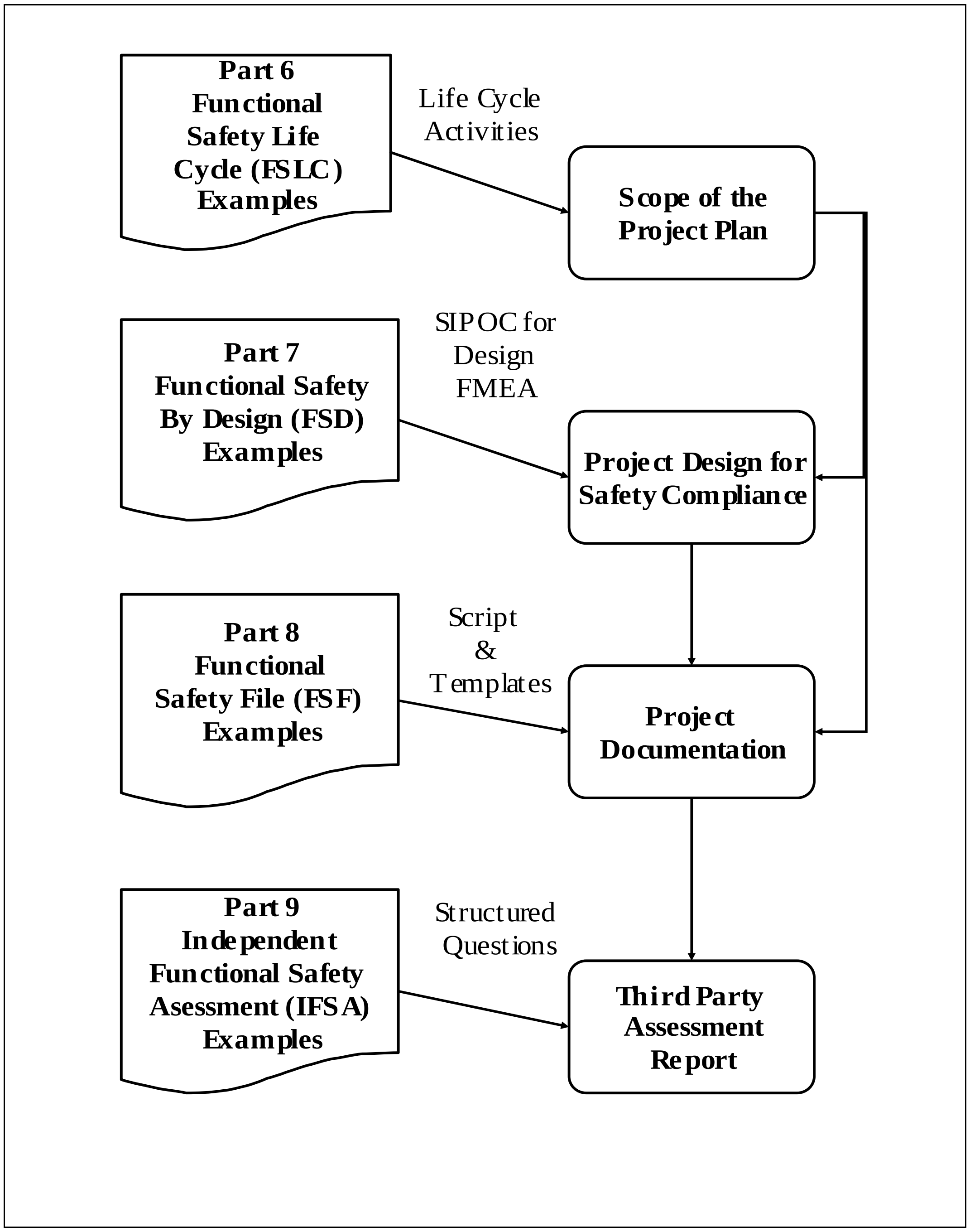
Part 6 Functional Safety Life Cycle (FSLC) Examples
Life Cycle Activities
Scope of the Project Plan
Part 7 Functional Safety By Design (FSD) Examples
SIPOC for Design FMEA
Project Design for Safety Compliance
Part 8 Functional Safety File (FSF) Examples
Script & Templates
Project Documentation
Part 9 Independent Functional Safety Asessment (IFSA) Examples
Structured Questions
Third Party Assessment Report

Figure 2 - Relationships among Parts 6, 7, 8, and 9

Part 7 – Additional Guidance: Functional Safety by Design (FSD) Examples

Part 7 bridges theory with practice for design activities by illustrating a Functional Safety Analysis (FSA) for person locator functions embedded in the DKYS components. The illustration addresses the conduct of a Job Hazard Analysis (JHA), a Hazard Analysis (HA), a Design Failure Modes and Effects Analysis (Design FMEA), and a Risk Analysis (RA). The report also references tools for conducting a Design FMEA.

Part 8 – Additional Guidance: Functional Safety File (FSF) Examples

Part 8 – Additional Guidance: Functional Safety File (FSF) Examples provides a prototype FSF Document Management System (DMS). Screen shots from the DMS define how a FSF may be organized and accessed. The prototype FSF-DMS supports preparation and management of FSF documents that would be submitted for an IFSA. The FSF-DMS uses the hypothetical next generation electronic safety equipment product, code-named DKYS, for Device that Keeps You Safe for illustration. Saros Inc's PDF Director System was used for rapid prototyping of the FSF-DMS. Appendix A provides information on PDF Director and other potential tools for DMS development.

Part 9 – Additional Guidance: Independent Functional Safety Assessment (IFSA) Examples

Part 9 – Additional Guidance: Independent Functional Safety Assessment Examples provides an approach to conducting an IFSA and an example audit questionnaire. The approach involves inspecting FSF documents using the questionnaire.

Intended Scope of Application

Systems, protection layers, and devices using electronics and software embedded in or associated with a PPE are within the intended scope of application. These provide

- Sensing and measuring biological, chemical and environmental characteristics of the site zone
- Providing auditory, vibration, visual, and sensory cues to an emergency responder
- Sensing and measuring physiological parameters about the emergency responder

- Identifying the location of the emergency responder
- Transmitting and receiving information about the site zone and the emergency responder
- Integrating and displaying safety information about site zones

Intended Users

The guidance is intended for use by life safety professionals and equipment manufacturers including:

- Manufacturers of components, subassemblies, and assemblies
- Final equipment manufacturers
- Systems integrators and installers
- Standards developers
- Equipment purchasers/users

Relevance of the Guidelines

- These recommendations do not supersede federal or state laws and regulations or recognized consensus standards.
- These recommendations are not equipment or application-specific.
- These recommendations do not serve as a compliance document.

Reference Guidelines and Standards

Mining industry guidelines prepared by NIOSH, MSHA and the mining industry manufacturers and entitled *Programmable Electronic Mining Systems: Best Practices Recommendations (in Nine Parts)* serves as a basis for these guidelines. Table 2 lists the published documents that form part of the mining industry guidelines. These documents can be found at http://www.cdc.gov/niosh/mining/topics/topicpage23.htm.

The mining guidelines are based on the requirements in existing standards—two of which are particularly applicable to PPE. These standards are the *ANSI UL 1998, Standard for Safety: Software in Programmable Components and IEC 61508, Functional Safety: E/EE/PE Safety-Related Systems*. Table 3 provides an overview of both standards.

9456	Part 1: 1.0 Introduction	John J. Sammarco, Thomas J. Fisher, Jeffrey H. Welsh, and Michael J. Pazuchanics	April 2001
9458	Part 2: 2.1 System Safety	Thomas J. Fisher and John J. Sammarco	April 2001
9460	Part 3: 2.2 Software Safety	Edward F. Fries, Thomas J. Fisher, and Christopher C. Jobes, Ph.D.	April 2001
9461	Part 4: 3.0 Safety File	Gary L. Mowrey, Thomas J. Fisher, John J. Sammarco, and Edward F. Fries	May 2002
9464	Part 5: Independent Functional Safety Assessment.	John J. Sammarco and Edward F. Fries	May 2002

Table 1 - Mining Industry Guidelines

STANDARD	**ANSI UL 1998**	**IEC 61508**
Title	Standard for Safety: Software in Programmable Components	Functional Safety: E/EE/PE Safety-Related Systems
Convened	1988	Early eighties
Approach	• Components • Embedded electronics and software • Integrated safety controls • Risk reduction based on coverage of identified hazards • Equipment safety requirements	• Components and systems • Networked • Separately instrumented safety systems • Risk reduction based on safety integrity level requirements • Equipment safety requirements
Standards Development Organization	Underwriters Laboratories (UL)	IEC SC 65A Working Group 9 and 10
Publication Date	First Edition: 1994 ANSI Second Edition: 1998	1998–2000
Where to obtain	http://www.comm-2000.com	http://www.iec.ch
Relevant URLs	http://www.ul.com/software/ http://www.ul.com/software/ansi.html	http://www.iec.ch/61508
Applications	UL 325, UL 353, UL 372, UL 1699, UL 1740, UL 2231, UL 61496	IEC 61511, IEC 62061, IEC 61496, IEC 61800-5

Table 2 - Overview of ANSI UL 1988 and IEC 61508

ACKNOWLEDGEMENT

In 1999, at the request of Congress, the National Institute for Occupational Safety and Health (NIOSH) established the National Personal Protective Technology Laboratory (NPPTL). The NPPTL provides leadership in the prevention and reduction of occupational disease, injury, and death for those workers who rely on personal protective technologies. Additional information about NPPTL can be found at http://www.cdc.gov/niosh/npptl and in NIOSH Publication 2003-127, National Personal Protective Technology Laboratory or by contacting Mr. Tim Rehak, the Project Officer at (412) 386-6866.

ABSTRACT

Emergency responders risk their lives to save the lives of others. It is a priority to provide them with the best equipment and the best guidance to minimize their exposure to hazards.

Advanced Personal Protective Equipment (PPE) incorporates product-ready technology in electrical, electronic, and programmable electronics. Use of newer materials, software, and wireless communications reduce safety risks. Experience has shown though, that these personal protective technologies may fail in ways not previously anticipated. Therefore, guidance for their use and integration is necessary.

The report, Part 3- Functional Safety by Design (FSD), is the third part in a nine-part series of recommendations addressing the functional safety of advanced personal protective equipment and systems (PPE) for emergency responders. Part 3 details best practice design criteria for use by manufacturers of PPE. Functional safety seeks to design safety into the equipment for all phases of its use. Electronics and software are components; therefore, design of these components must take into account the overall achievement of functional safety. Part 3, Functional Safety by Design (FSD) provides best practice design criteria for use by manufacturers of PPE.

1.0. INTRODUCTION

1.1. Background

The PPE industry is using electronics and software technology to improve safety of emergency responders and to increase the likelihood of survival of victims. Electronics and software now provide protection, monitoring, and communication functions for emergency responders. Although use of electronics and software provides benefits, it also adds a level of complexity that, if not properly considered, may adversely affect worker safety.

Failure of functionality embedded in electronics and software may lead to new hazards or worsen existing ones. Electronics and software have unique failure modes that may be different from mechanical systems or hard-wired electronic systems. The situation led to the development of criteria for designing functional safety into the entire system from initial conceptualization to retirement.

Functional safety seeks to design safety into the equipment for all phases of its use. Software is a sub-system; therefore, software safety is part of functional safety. .

Part 3 introduces the design elements used to achieve functional safety for electronics and software used in emergency responder applications. Section 2 presents the overall design approach; Section 3 presents the detailed electronic hardware and software design criteria and Section 4 presents the detailed analysis and test criteria. These design criteria are intended to be used by the life safety equipment manufacturers including component manufacturers, sub assembly manufacturers, and final equipment manufacturers; systems integrators and installers; and life safety professionals when designing equipment and developing safety design and performance standards.

Part 3 describes best practices to prevent, remove, and recover from both random and systematic failures.

1.2. The As Low As Reasonably Practical (ALARP) Concept

Innovative designs of PPE using electronics and software technology have more embedded safety functions and an increased number of interfaces. These innovations provide more life safety features for the emergency responder as well as an enhanced ability to respond to complicated threat scenarios. A primary objective then is to achieve an acceptable level of risk that is as low as reasonably practical (ALARP) as shown in Figure 3.

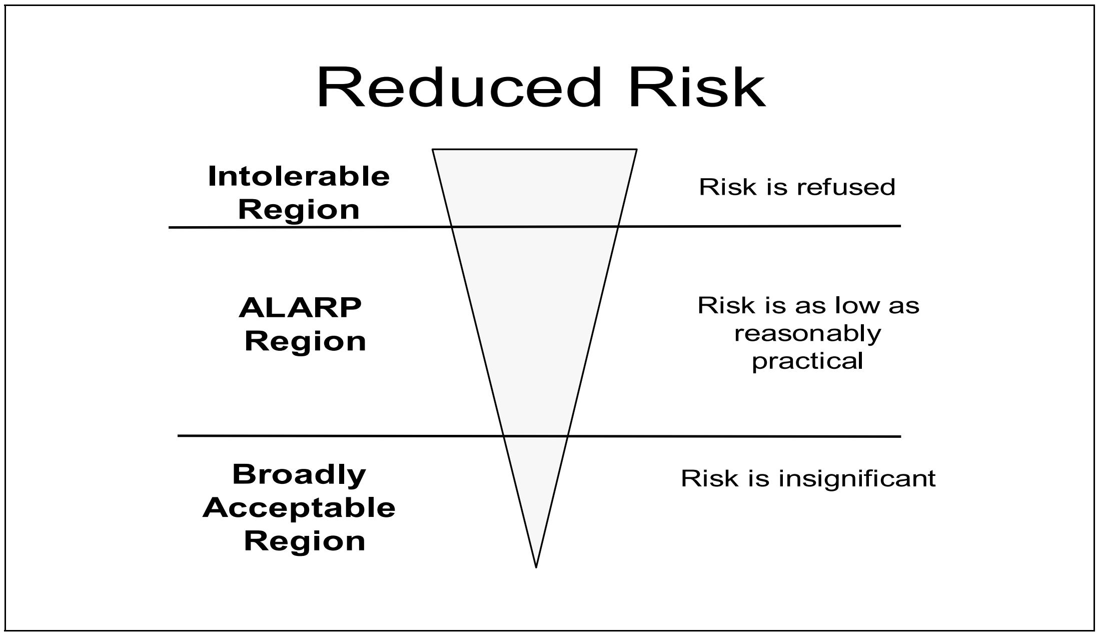

Figure 3 - The ALARP concept.

The ALARP concept drives the selection of tools for reducing risk and hence achieving functional safety in PPE. Additional discussion about this concept as it applies to PPE electronics and software may be found in Part 1.

1.3. Safety by Design Considerations for PPE

1.3.1. Random and Systematic Failures

To achieve functional safety by design for PPE, the product design engineer defines functional safety and performance requirements that address both the potential for failures due to random phenomena and failures due to design or systematic phenomena.

Operating environments contribute to random failure of electronic components by affecting electronic function. Interference from outside sources, such as electromagnetic emissions; temperature extremes; humidity, moisture, and water exposure; heat and flame exposure; chemicals, dust, and debris, and extreme impacts can corrupt electronically maintained data and software instruction processing.

Systematic failures result from inadequacies in the design and logic. These include design errors such as:

- Incorrect software algorithms and interfaces,
- Coding errors, including syntax, incorrect signs, endless loops and the like,
- Timing errors that can cause program execution to occur prematurely or late or not at all,
- Latent errors not detectable until a given set of conditions occur, and
- Failure of the PPE to perform any function at all

Systematic failures affect functional safety in two ways: 1) output values and/or timing that permit the system to reach a state that could lead to a mishap and 2) failure to identify or properly handle events that if not responded to, could lead to a mishap.

1.3.2. Reducing Risk by ALARP Practices

To address random and systematic failures of the PPE, the product engineer focuses on narrowing the middle section of the ALARP region by implementing design and test practices to reduce risk. See Figure 4. This involves applying fault prevention, removal, and detection and recovery practices.

An example of a fault prevention practice used during coding is to make sure that all structures in the code that store data are set to a value before they are used. This is known as a ‘set before use’ coding requirement. For example, if a data structure stores temperature values, it must be initialized to a value before that data can be processed and provided to the user of the PPE.

Decision coverage testing is an example of a fault removal activity conducted during software design. Decision coverage testing executes all decision paths in the code at least once. Consider the situation where a decision is encoded to annunciate a beep in a PPE device when there is no movement of a emergency responder. To encode this decision, what is considered movement and no movement would be defined by a mathematical algorithm, which is then encoded in the software. Decision coverage testing would consider the definitions of no movement versus movement, and the tester would review the code and execute the logic for the two decision paths in the code to

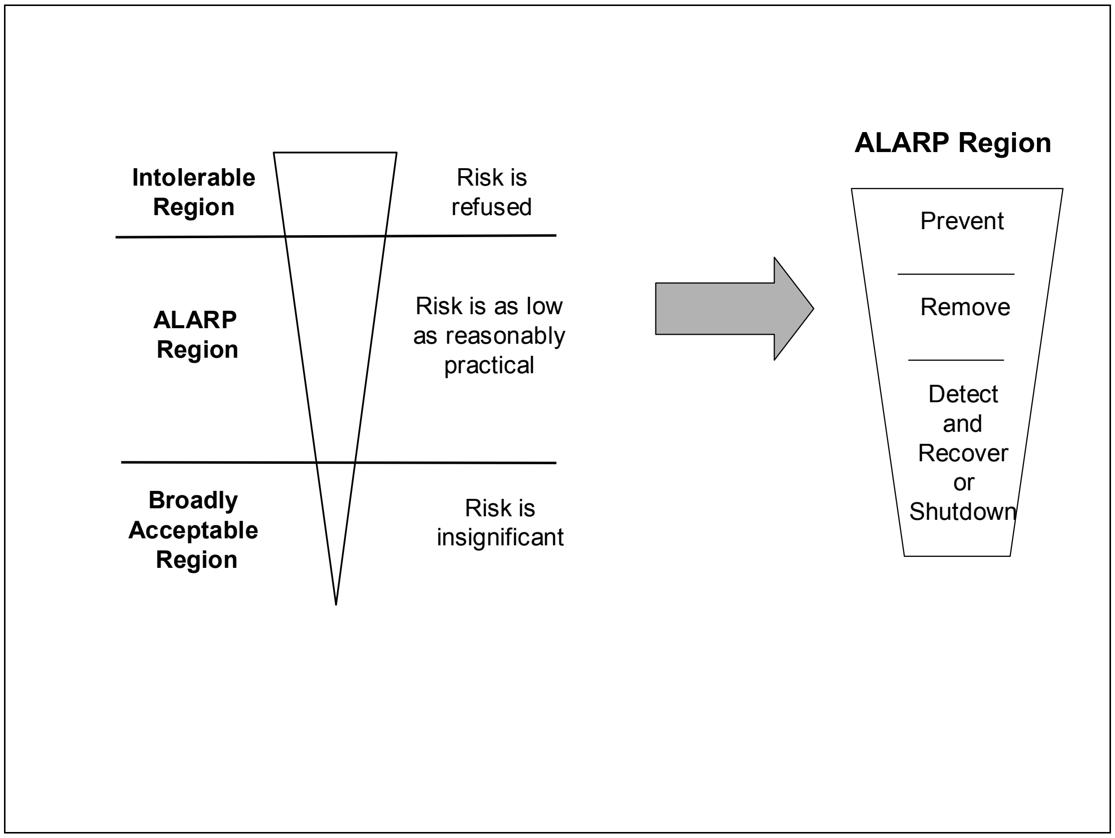
Intolerable Region
Risk is refused
ALARP Region
Risk is as low as reasonably practical
Broadly Acceptable Region
Risk is insignificant
ALARP Region
Prevent
Remove
Detect and Recover or Shutdown

Figure 4 - ALARP region activities.

detect if there were any logic faults or “bugs” along each of these paths. If what is considered no movement vs. movement is based on input such as a emergency responder’s weight set by the emergency responder before use, then the tester would repeat decision coverage testing multiple times within different carefully selected weights. If the software developer had incorporated logic that inconsistently considered the decision to beep when movement was very rapid, the “bug” could be identified and removed. It is important to note that functional or certification testing may include a “very rapid movement" test that would test for carefully selected weights. However, the inconsistent logic may be processed only when a weight was entered incorrectly and the user re-entered the weight in a certain way. If the functional or certification testing did not consider the specific way the user operated the device, the bug would not be detected. Additionally, finding the inconsistent logic during decision coverage testing, which is applied earlier in the life cycle, would be less costly to fix.

An example fault detection method would be to compute and send an error correcting code when transmitting data. This retransmission may result in functional recovery of the PPE if there is enough time to retransmit the data when an error occurs. It could also result in shutting down the PPE if the data could not be retransmitted successfully due to interference from other transmission sources. In both cases, the PPE would be designed to reduce risk to as low as reasonably practical by providing as much functional capability as appropriate for the emergency responder's safety as defined by fitness for use criteria. The designer may thus permit retransmission several times within an acceptable period and, if not successful, the receiving device would warn the user of transmission problems and either shutdown or provide only safety functions that either use data history or do not rely on transmitted data. The designer could also further reduce risk by using the receiving device to periodically check for successful signal transmissions.

2.0. PPE SYSTEM DESIGN APPROACH AND DESIGN CRITERIA

2.1. Objectives:

2.1.1. Eliminate hazards through design.

2.1.2. Reduce risk to as low as reasonably practical.

2.2. Recommendations

2.2.1. Conduct a hazard analysis for the PPE electronics and software system to identify hazards. (See Table 3 and Appendix A for summary descriptions of these methods including references).

2.2.2. Conduct a risk analysis that assigns a risk reduction factor (RRF) and that considers other layers of protection for the safety functions provided by the electronics and software. (See Table 4 and Appendix B. for summary descriptions of these methods including references.)

2.2.3. Begin the hazard and risk analysis activities at the safety requirements definition phase of the project and continue until the PPE is decommissioned.

2.2.4. Carefully analyze functional safety impact when changing PPE components from other technologies to electronics and software.

2.2.5. Use other proven non-electronic devices as the primary layer of protection of safety with electronics and software implemented functions as secondary layer of protection, as appropriate

2.2.6. Minimize design complexity

2.2.7. Increase the reliability of electronic component performance through the use of diagnostics, redundancy and error recovery techniques, as appropriate

2.2.8. Select electronic components which have an acceptable failure rate over the expected operational life of the equipment.

2.2.9. Design for protection of the electronics and software against the impact of temperature, humidity, water impingement, electromagnetic interference (EMI), shock, vibration, and contaminants

2.2.10. Specify input and output descriptions that include data representation information and acceptable data values

2.2.11. Precisely define what are safe states

2.2.12. Describe how the safety functions are organized and assigned to electronics and software

Hazard Analysis Method	Description
Preliminary Hazard Analysis (PHA)	An analysis technique used in the early conceptual stages of design and development. Typically, a team is used to identify potential hazards of the PPE and its major components including electronics and software.
Failure Modes and Effects Analysis (FMEA)	This analysis identifies failures of electronics and software/firmware components and integrated subsystems and their effects on the system. In essence, a "bottom up" approach starting with the systems' components.
Hazard and Operability Studies (HAZOP)	A systematic and structured qualitative method of study conducted by a multiple disciplinary team. Guidewords are applied to various parameters to stimulate thinking concerning possible deviations.
Fault Tree Analysis (FTA)	A logical "top down" method of structuring events and failures leading to a hazard. It is a logical method of deduction utilizing a graphical depiction of events.
Event Tree Analysis (ETA)	A logical, bottom-up graphical technique to determine outcomes from a single initiating hazardous event.
Potential or Predictive Human Error Analysis	A team-based method similar in concept to HAZOP, however, this analysis focuses on human tasks and the associated error potential.
Operating and Support Analysis (O&SA)	OS&A seeks to identify hazards during operation and maintenance, find the root causes, determine the acceptable level of risk, and recommend risk reductions.
Action Error Analysis (AEA)	AEA is used to identify operator errors and the subsequent operations.
Interface Analysis	Interface analysis is used to identify hazards resulting from physical, functional, logical, and time-based interface inconsistencies. Interface analysis includes addressing the human-computer interface.
Sequentially-Timed Events Plot (STEP)	STEP is an event-driven approach to define systems, analyze operation, and to investigate mishaps. STEP is an analytical approach that graphically depicts sequentially timed events.

Table 3 - Example hazard analysis methods

Risk Analysis Methods	Description
Hazardous Event Severity Matrix	Risk is determined by using severity, frequency and other layers of protection.
Layers of Protection Analysis	Determines risk when multiple protection layers serve to reduce risk to an acceptable level.
Risk Graph	Risk is determined by using severity and frequency plus the possibility of avoiding danger and the probability of the unwanted occurrence.
Risk Matrix	Risk is determined by using severity and frequency.

Table 4 – Risk Analysis Methods

2.2.13. Trace the end-to-end implementation of the safety function across analog and digital interfaces, product interfaces, and electronics and software components

2.2.14. Document mathematical algorithms used to accomplish the assigned functionality including assumptions and degenerate conditions

2.2.15. Specify processing performance parameters for each safety function, such as:

- Timing deadlines, especially real-time deadlines and timing requirements for interrupt services
- Scheduling constraints including task priorities
- Required processing resources (i.e. memory size, type of memory, and processor speed)
- Constraints on resource allocations (e.g. amount of memory, processor speed)
- Sensitivity of electronic performance to changes in safety parameter values and resource allocations
- Expected throughput

- Effects on throughput (i.e. operating systems performance overhead, fault detection and recovery overhead, resource contention, locality of memory, task prioritization, and memory addressing schemes)

2.2.16. Consider using layers of protection based on other technologies to minimize the effects of a PPE failure after it occurs, such as:

- Provide separate warning devices: Use means to detect the condition and produce an adequate warning signal to alert personnel of the hazard. Warning signals and their application minimize the probability of incorrect personnel reaction to the signals and are often standardized within like types of systems.

Note: What is considered an adequate warning signal would be specified as part of the safety requirements for the application. For example, "The warning signal shall be 10-15 decibels above the ambient level of any masking noise to ensure that it can be heard".

- Develop procedures and training: Use procedures and training where it is impractical to eliminate hazards through design selection or adequately reduce the associated risk by using other layers of protection. Avoid using warning, caution, or other written advisory as the only risk reduction method.

3.0. DETAILED ELECTRONICS AND SOFTWARE DESIGN CRITERIA

3.1. Objectives

3.1.1. Consider best practices when designing the electronics and software.

3.1.2. Consider applying the requirements of ANSI UL 1998 and IEC 61508 when designing safety functions to be implemented using programmable electronics and software technology.

3.2. Recommendations

3.2.1. Apply physical and functional separation to both electronics and software.

3.2.2. Avoid having unnecessary or unused electronics and software features, functions, or components in the PPE.

3.2.3. Minimize data and control flow complexity

3.2.4. When incorporating safety interlocks:

- Restrict them from intentionally or inadvertently being in a bypass or an override state during normal operation.
- Consider human factor principles such that they are not unduly difficult or uncomfortable. [4]

3.2.5. When reusing electronics and software component or subsystems, revisit the entire FSLC to determine what design and test requirements are needed.

3.2.6. When electronics and software are used to implement multiple safety functions, then consider the RRF category as the category for the safety function that provides the most risk reduction.

3.2.7. When electronics and software are used to implement safety and non-safety functions, treat the entire implementation as a safety function.

3.2.8. Consider the following power-up/power loss criteria [NATO 1997][5].

- Power up in a defined safe state
- Upon power-up, use diagnostics to verify the PPE is in a safe state and that communications with other PPE systems are working properly.
- Place the PPE system in a safe state during and after power-up, power loss, and intermittent power faults/interruptions.

3.2.9. Consider the following mode transition criteria:

---

[5] Safety requirements and guidelines for munitions related safety-critical computing systems. draft NATO standardization agreement (STANAG) 4404, edition 1, Document AC/310-D/139.

- Clearly identifying modes and mode transitions (e.g. manual, automatic, remote)
- The potential safety impact of mode transitions during system operation, repair, and maintenance.
- Requiring mode transitions to have human initiation and acknowledgment
- Providing warning indications, for example auditory and visual arms, before transitioning to another mode
- Providing a capability for overriding automatic operation during emergencies
- When overriding automatic control to prevent a hazard consider requiring a single action that is readily identifiable and unambiguous

3.2.10. Use robust software design and defensive programming techniques, such as

- Setting all variables to initial values before use by any instruction
- Use program pre and post assertions
- Handle exceptions
- Preventing, detecting, and resolving non-terminating and nondeterministic states and error states, such as undefined branch conditions, zero division, and underflow and overflow
- Provide in-line code documentation with explanatory comments and assumptions

3.2.11. Address the Critical and Supervisory Sections of the Software

- Identify and document which portions of the software are safety critical and which are not
- Keep the safety critical portions as small, simple, and concise as possible.
- Initialize software to a documented safe state
- Isolate both physically and logically all critical and supervisory sections of the software from the non-critical sections.

NOTE 1: If it is not possible to isolate the software, all of the software is considered critical.

- Employ means to avoid or to detect and recover from memory usage and addressing conflicts.
- Code the supervisory section to maintain control of the execution of the software at all times during the operation of the PPE
- Initiate a fail-safe or fail-operational procedure if a fault occurs in a critical or supervisory section.
- Employ means such as error checking and/or correcting procedures to preserve the integrity of data and instructions used by critical and supervisory sections.

3.2.12. Consider the following practices for the Software Interface to the Equipment :

- When equipment initialization is allocated as a software function, the software should initialize the equipment to a documented safe state.
- Whenever the software terminates, the equipment should maintain a documented safe state.
- Any procedure or instruction intended to halt the software processing should maintain the equipment in a safe state.

3.2.13. Consider the following practices for the User Interface

- The operator interface is a safety function.
- Consider usability criteria when developing the user interface[6]
- Preventing accidental operation that could lead to a hazard
- Requiring a single action by the operator when canceling an operation.
- Placing the PPE in a safe state when data is input incorrectly or in an

[6] D.Hix and H.R. Hartson, Developing User Interfaces: Ensuring Usability through Product & Process, New York: Wiley and Sons, 1993. ISBN 0471578134

unacceptable sequence or processing is cancelled

- When two or more operator inputs are used to initiate an operation that could lead to a hazard, provide confirmation of the operation
- Defining and documenting all interactions between the operator and the PPE
- Providing the following information through the operator interface:
  - The process sequence
  - The current mode (i.e., automatic, manual or abnormal)
  - Degraded operation due to a failure
  - Alerts to any safety function bypass
- Making interfaces for maintenance/diagnostic purposes accessible, physically and visually, such that personnel are not exposed to a situation that could lead to a hazard
- Disallowing users from modifying any safety application electronics and software
- Time limits and other parameters of the software should not be changeable by a user in such a way that the intended execution of critical and supervisory sections is adversely affected.
- The time limits and other parameters of the software that are intended to be configured by qualified service personnel should be prevented from being changed to the extent that the intended operation of the critical or supervisory sections of software is adversely affected.
- The software should require two or more user responses to initiate an operation that is capable of resulting in a risk.
- Input commands that are capable of resulting in a risk when executed should not be initiated without operator intervention when those commands are received from an external source and should require two or more user inputs.
- Incorrect input should not adversely affect execution of critical sections of software.

3.2.14. Choice of Software Language

- Consider languages that provide safe computing constructs.

3.2.15. Design to address programmable electronic device hardware failures as shown in Table 5 - Characteristic faults of electronic components and further described in Appendix C.

COMPONENT		**FAULT**
CPU	Register, internal RAM	Stuck-at for data and addresses; DC model for data and addresses; Dynamic cross-over for memory cells; No, wrong, or multiple addressing
	Coding and execution including flag register	Wrong coding or no execution No definite failure assumption
	Address calculation	Stuck-at DC model No definite failure assumption
	Program counter, stack pointer	Stuck-at DC model
Bus	General	Stuck-at of the addresses; Time out
	Memory management unit	Stuck-at of data or addresses; Wrong address decoding
	Direct memory access	No or continuous access; DC model for data and addresses; Wrong access time
	Bus-arbitration	Stuck-at of arbitration signals No or continuous or wrong arbitration
Interrupt handling		No or continuous interrupts; Cross-over of interrupts
Clock (Quartz)		Sub- or super harmonic
Invariable memory		Stuck-at for data and addresses; DC model for data and addresses; All faults which affect data in memory
Variable memory		Stuck-at for data and addresses; DC model for data and addresses; Dynamic cross-over for memory cells; No, wrong or multiple addressing
Discrete hardware	Digital I/O	Stuck-at; DC model, Drift and oscillation

	Analog I/O	Stuck-at;
		DC model;
		Drift and oscillation

	Power supply	Stuck-at; DC model; Drift and oscillation
Communication and mass storage		Wrong data or addresses; No transmission ; All faults which effect data in the memory; Wrong transmission time; Wrong transmission sequence

Table 5 - Characteristic faults of electronic components

Note 2: Bus-arbitration is the mechanism for deciding which device has control of the bus. "stuck-at" is a fault category, which can be described with continuous "0" or "1" or "on" at the pins of a component. "DC model" (DC = direct current) denotes stuck-at faults, stuck-open, open or high impedance outputs as well as short-circuits between signal lines.

- Provide fault detection during all modes of operation, especially for startup, shutdown, and unintended mode transitions.
- Upon detection of a fault that could lead to a hazard, provide a fault tolerance capability for maintaining and achieving the safest state possible.
- Use sufficient fault tolerance to maintain the RRF for the safety function being provided.
- Use watchdog timers with a separate time base to monitor the behavior and plausibility of computer operation and program sequencing.
- Integrate the execution of fault detection functions with normal system operation.

NOTE 3: Unsafe states may not result from execution of these functions

- Offline detection of faults (i.e., running of diagnostic tests when the PPE is not in an operating mode) may not lead to an unsafe state.
- Use diagnostic tests to identify failures to the level of a field replaceable or repairable module.
- Detect failures of all electronics and microelectronic components on start-up and during operation.

3.2.16. Shared Devices

- Where the device is shared among more than one PPE component, design the PPE so that failure of a PPE component does not degrade the performance of the other PPE components using by that device

3.2.17. Smart devices

- Smart Devices (i.e., sensors, valves, transmitters with built-in diagnostics) are recommended for improving diagnostics and reliability.

NOTE 4: For example, a sensor with an analog output provides more diagnostic capabilities than a discrete on/off limit switch. Diagnostic checks of range, rate, and data plausibility are possible with analog output.

3.2.18. Electrical Interfaces

- Electrical cable or connector failures must not place the PPE in an unsafe state.

4.0. DETAILED ELECTRONICS AND SOFTWARE ANALYSIS AND TEST CRITERIA

4.1. Objectives:

4.1.1. Consider best practices when analyzing and testing the electronics and software.

4.1.2. Consider applying the requirements of ANSI UL 1998 or IEC 61508 when analyzing and testing safety functions to be implemented using programmable electronics and software technology.

4.2. Recommendations

4.2.1. Analyze possible combinations of electronic hardware failures, software faults, and other events that are capable of resulting in a risk will be conducted.

4.2.2. Test the electronics and software under the following operating conditions:

- Environmental (i.e., temperature, moisture, dust, and vibration)
- Electrical (i.e., EMI, power sources, supply voltages, and data signals)
- Physical (i.e., rate and range of movement)
- Logical (i.e., conditional responses based on Boolean expressions)
- Temporal (i.e., clock times, response times, and delay times)

4.2.3. Evaluate the reliability of the electronics components and subsystems for their ability to surpass a maximally accepted failure rate during stressed operating conditions such as those identified in 4.2.2.

4.2.4. Analyze and test the software in accordance with the RRF categories as recommended in Table 6 and further described in Appendix C.

5.0. SUMMARY

The PPE industry is using electronics and software technology to reduce life safety risks for emergency responders and victims. Electronics and software have failure modes that differ from mechanical systems or hard-wired electronic systems. The failure modes result from random phenomena (i.e., electromagnetic emissions; temperature extremes; humidity, moisture, and water exposure; heat and flame exposure; chemicals, dust, and debris, and extreme impacts). The failure modes may result as well from systematic (or logic) errors (i.e., inconsistent software algorithms and interfaces, coding errors, timing errors, latent errors, and failure of the PPE to perform any function at all). To achieve safety requires a system design approach addressing hardware, software, human behavior, and the operating environments over the equipment's life cycle.

Functional Safety by Design details best practice recommendations for use by manufacturers of PPE. Recommendations address adding design for safety activities including safety specification, hazard analysis, hardware and software diagnostics, safety-focused reviews and tests, specification to test traceability, safety documentation, and training. By referencing these criteria during product design, the product designer works toward reducing the risk of product functions failing in a way that may result in a hazard.

SOFTWARE ANALYSIS AND TEST METHOD	**RRF 3** *Non-hazardous, Non-fire environment*	**RRF 2** *Hazardous or Potentially Hazardous Non-Fire Environment*	**RRF 1** *Severe Exposure Fire Environment and Potential for Fire*
Reviews, Walkthroughs, and Inspections:			
Specifications/requirements review	Highly recommended	Highly recommended	Highly recommended
Design review	Highly recommended	Highly recommended	Highly recommended
Code review/reading	Highly recommended	Highly recommended	Highly recommended
Mathematical proofs	----	----	Recommended
Structure analysis	----	Highly recommended	Highly Recommended.
Error and anomaly analysis	----	----	Highly recommended
Implementation Based Testing:			
Statement coverage	Highly recommended	Highly recommended	Highly recommended
Branch coverage	Recommended	Highly recommended	Highly recommended
Multiple condition coverage	----	Highly Recommended	Highly recommended
Decision-to-decision path	----	----	Highly recommended
Linear Code Sequence and Jump testing	----	----	Highly recommended
Data flow coverage	----	recommended	Highly recommended
Specification-Based Testing:			
Interface testing	Recommended	Recommended	Highly Recommended
Usage testing	----	----	Highly recommended
Equivalence partitions	Highly recommended	Highly recommended	Highly recommended
Boundary values	Highly recommended	Highly recommended	Highly recommended
Special values	Highly recommended	Highly recommended	Highly recommended

Table 6 - Recommended software analysis and test methods by RRF Factor

6.0. ABBREVIATIONS

ABBREVIATION	DEFINITION
ALARP	As Low As Reasonably Practical
ANSI	American National Standards Institute
CMM	Capability Maturity Model
CTQ	Critical to Quality
DFMEA	Design Failure Modes and Effects Analysis
DKYS	Device that Keeps You Safe
DMS	Document Management System
EIA	Electronic Industries Alliance
EMI	Electromagnetic Interference
ESE	Electronic Safety Equipment
ETA	Event Tree Analysis
FMEA	Failure Modes and Effects Analysis
FSA	Functional Safety Analysis
FSD	Functional Safety by Design
FSF	Functional Safety File
FSLC	Functional Safety Life Cycle
FSLC-PMT	Functional Safety Life Cycle – Project Management Template
FTA	Fault Tree Analysis
HA	Hazard Analysis
HAZOP	Hazard and operability study
IAFF	International Association of Fire Fighters
IDLH	Immediately Dangerous to Life and Health
IFSA	Independent Functional Safety Assessment
IEC	International Electrotechnical Commission
IPL	Independent Protection Layer
JHA	Job Hazard Analysis
LOPA	Layer Of Protection Analysis
MOC	Management Of Change

ABBREVIATION	DEFINITION
MSHA	Mine Safety and Health Administration
NFPA	National Fire Protection Association
NIOSH	National Institute for Occupational Safety and Health
NPPTL	National Personal Protective Technology Laboratory
OSHA	Occupational Safety and Health Administration
PASS	Personal Alert Safety System
PDA	Personal Digital Assistant
PFD	Probability Of Failure On Demand
PHL	Preliminary Hazard List
PM	Project Manager
PPE	Personal Protection Equipment
QMS	Quality Management System
RA	Risk Analysis
RFI	Radio Frequency Interference
RFID	Radio Frequency Identification
RPN	Risk Priority Number
RRF	Risk Reduction Factor
SEI	Software Engineering Institute
SFTA	Software Fault Tree Analysis
SIL	Safety Integrity Level
SLC	Safety Life Cycle
SIPOC	Supplier-Input-Process-Output-Customer
SLC	Safety Life Cycle

7.0. GLOSSARY

As low as reasonably practical (ALARP): A risk level associated with failure of the PPE that is considered acceptable because it is as low as reasonably practical.
Balanced Scorecard: Method for measuring organizational success by viewing the organization from customer, financial, internal business process, and learning and growth perspectives

Component: Any material, part, or subassembly used in the construction of PPE. Computer hardware and software are components of PPE.

Configurability: The ability to rapidly configure a PPE system to meet different life safety threats and to account for different user needs.

Compatibility: Requirements for the proper integration and operation of one device with the other elements in the PPE system.

Critical to Quality Tree: A six sigma method that uses a tree diagram for identifying important characteristics of a process or product that is critical to quality

Electronic Safety Equipment: Products that contain electronics embedded

in or associated with the product for use by emergency services personnel that provides

enhanced safety functions for emergency services personnel and victims during emergency incident operations (from NFPA 1800).

Failure modes and effects analysis (FMEA): This technique uses deductive logic to evaluate a system or process for safety hazards and to assess risk. It identifies the modes in which each element can fail and determines the effect on the system.

Functional Safety of ESE: ESE that operates safely for its intended functions.

Functional Safety Analysis: The process of identifying failures which lead to missed or inaccurate delivery of functions causing the potential for harm.

Functional safety by design (FSD): A system design approach that involves looking at the entire context of use for the equipment or system, identifying hazards, designing to eliminate or reduce hazards, and doing this over the entire life cycle for the PPE.

Functional safety file (FSF): Safety documents retained in a secure centralized location, which make the safety case for the project.

Functional safety life cycle (FSLC): All activities conducted in accordance with a functional safety approach to designing and building safety into the entire system from initial conceptualization to retirement.

Hazard: An environmental or physical condition that can cause injury to people, property, or the environment.

Hazard and operability study (HAZOP): This is a systematic, detailed method of group examination to identify hazards and their consequences. Specific guidewords are used to stimulate and organize the thought process. HAZOP [Ministry of Defense 1998] has been adapted specifically for systems using programmable electronic systems (PES).

Hazard Analysis: The process of identifying hazards and analyzing event sequences leading to hazards.

Hazard and risk analysis: The identification of hazards, the process of analyzing event sequences leading to hazardous events, and the determination of risks associated with these events. Risk analysis determines the risk reduction requirement for the equipment or system based on qualitative or quantitative approaches**.**

Hazard and risk analysis team: The group of emergency responders, electrical, electronics, computer hardware/software, manufacturing, and safety specialists responsible for the safety and integrity evaluation of PPE from its inception through its implementation and transfer to operations to meet corporate safety guidelines.

Hazard List: A list used to identify for tracking hazards throughout the FSLC. The list describes each hazard in terms of the event (s) that would lead to an accident scenario. When the hazard is identified during an accident analysis, the description of the hazard will also reference the accident scenario and consequences and measures that may be taken to avoid or prevent recurrence. The hazard list is used as input to the FMEA.

Human-computer interaction: The application of ergonomic principles to the design of human-computer interfaces.

Human-machine interface: The physical controls, input devices, information displays, or other media through which a human interacts with a machine in order to operate the machine.

Independent department: A department whose members are capable of conducting an IFSA. The department must be separate and distinct from the departments responsible for the activities and subject to Functional Safety Assessment or validation, taking place during the specific phase of the FSLC.

Independent functional safety assessment (IFSA): A systematic and independent examination of the work processes, design, development, testing, and safety file documentation for a product/machine/control system to determine compliance with applicable safety recommendations/standards/regulations.

Independent organization: An organization that is legally independent of the development organization whose members have the capability to conduct IFSAs. The organization member conducting the audit must be separate and distinct from the activities and direct responsibilities taking place during a specific phase of the overall FSLC that is subject to Functional Safety Assessment or validation.

Independent person: A person who is capable of conducting an IFSA. The person must be separate and distinct from the activities and direct responsibilities taking place during a specific phase of the overall FSLC that is subject to Functional Safety Assessment or validation.

Independent protection layer (IPL): Engineered safety features or protective systems or layers that typically involve design for safety in the equipment, administrative procedures, alarms, devices, and/or planned responses to protect against an imminent hazard. These responses may be either automated or initiated by human actions. Protection should be independent of other protection layers and should be user and hazard analysis team approved.

Internal assessment: Conducted by the manufacturer to determine that the design and development process continues to comply with the safety plans and the safety file procedures. A report is issued and reviewed by appropriate management personnel.

Interoperability: The ability of PPE equipment and systems to provide services to and accept services from other PPE equipment and systems and to use the services so exchanged to enable them to operate effectively together.

Layer of protection analysis (LOPA): An analysis that identifies risk reduction targets by evaluating selected risk scenarios.

Lean Manufacturing: Implementing steps to reduce waste during the manufacturing process. There are eight types of waste – defects, overproduction, waiting, unused talent, transportation, inventory, motion, and extra processing.

Maintainability: The ability to maintain a PPE with minimum maintenance and repair so that the PPE can remain in service with full operation.

Mishap: An unplanned event or series of events resulting in death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment.

Periodic follow-up safety assessment: A systematic, independent, and periodic assessment which determines if the functional safety of the PPE is maintained.

Personal alert safety system (PASS): Devices that sense movement or lack of movement and that automatically activate an audible alarm signal to alert others in locating an emergency responder.

Personal protection equipment (PPE): Equipment and systems that provide the following life-safety protection functions:

- Protection against thermal, abrasion, puncture wounds, respiratory, vision, hearing and limited chemical and biological pathogen exposure hazards
- Monitoring of physiological, chemical, biological, and environmental parameters
- Communication among emergency responders and between emergency responders and victims

PPE functional requirements: Functions provided by the application including those functions required to meet NFPA equipment safety requirements.

PPE performance requirements: Timing and resource constraints imposed by the application including constraints needed for safety performance, such as delivering data to the user within the time frame required.

Preliminary hazard analysis (PHA): This technique uses the results of PHL, lessons learned, system and component design data, safety design data, and malfunction data to identify potential hazard areas. In addition, its output includes ranking of hazards by severity and probability, operational constraints, recommended actions to eliminate or control the hazards, and perhaps additional safety requirements.

Preliminary hazard list (PHL): This is the first analysis performed in the system safety process and strives to identify critical system functions and broad system hazards. It uses historical safety data from similar systems and mishap/incident information hazard logs to guide the safety effort until more system-specific is developed.

Probability of failure on demand (PFD): A value that indicates the probability of a system failing to respond on demand. The average probability of a system failing to respond to a demand in a specified time interval is referred to as "PFD avg."

Project plan: A document that addresses the entire life cycle including development and use activities, management of change activities, and the documentation of safety. The project plan is updated throughout the life cycle.

Proven In Use: The component is considered reliable because it has been used in several products in the application over a period of time and reliability data is available for the component.

Random hardware failure: A failure, occurring at a random time, which results from one or more of the possible degradation mechanisms in the hardware

Rapid fire progression: A rapid rise in temperature that leads to an almost instantaneous combustion of materials over a larger area.

Record: Stating results achieved or providing evidence of activities performed.

Requirements Specification: A list of PPE requirements where each requirement is uniquely identified, traceable, and has safety performance criteria specified.

Retrospective Validation: Validation after the ESE has been fielded which is based on review of development documentation and testing and on field problem reports.

Risk analysis: Determination of the risk reduction requirement for the equipment or system based on qualitative or quantitative approaches.

Risk management summary: Details the risk management activities and summarizes the important risks identified and the means used to remove or mitigate them.

Risk reduction factor (RRF): Measure of the amount of risk reduced through implementation of safety equipment, training, and procedures. RRF is usually expressed as a reduction in the risk of loss of life.

Risk Priority Number (RPN): A number which establishes the priority for addressing the risk. RPN is computed based on severity, probability, and detectability. The higher the number obtained the higher the priority for addressing the potential failure.

Safety: Freedom from unacceptable risks.

Safety claims: A safety claim is a statement about a safety property of the PPE, its subsystems and components.

Safety integrity: The probability of a safety-related system satisfactorily performing the required safety functions under all the stated conditions within a specified period.

Safety Policy: A statement which describes in general the organizational commitment to safety and how safety issues will be addressed.

Safety statement: A succinct summary statement affirming the completeness

and accuracy of the FSF and the level of safety demonstrated for the PPE.

Safety life cycle (SLC): All activities conducted in accordance with a systems approach to designing and building safety into the entire system from initial conceptualization to retirement.

Scalability: The ability to scale up PPE to respond to threats, which cross jurisdictional boundaries.

Suppler Input Process Output Customer (SIPOC) Diagrams: Diagrams which show suppliers, the required input, the steps in a process, the output produced, and the customer of that output.

Systematic failure: A failure related to a certain cause, which can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation, or other relevant factors. Examples of systematic failures include design errors in interfaces and algorithms, logic/coding errors, looping and syntax errors, and data handling errors.

Traceability: Ability to trace the history, application or location of that which is under consideration.

Usability: Ease of use of the PPE. Usability is specified by stating performance requirements that define what users expect to accomplish.

Validation: Analysis, review, and test activities that establish that the PPE is built in accordance with the emergency responder needs. Did we build the right PPE?

Verification: Analysis, review and test activities that establish that the PPE is built in accordance with the PPE specifications. Did we build the PPE right?

Voice of the Customer (VOC): Six Sigma methods for collecting data on the desires and expectations of the customer. These methods include focus groups, surveys, websites, customer site visits, and interviews with distributors and/or retailers, current and lost customers.

APPENDICES

APPENDIX A - HAZARDS ANALYSIS

The objective of hazards analysis is to identify and analyze hazards and event sequences leading to hazards. Many techniques, ranging from simple qualitative methods to advanced quantitative methods, are available to help identify and analyze hazards. The use of multiple hazard analysis techniques is recommended because each has its own purpose, strengths, and weaknesses. The System Safety Analysis Handbook [SSS, 1999][7] provides extensive listings and descriptions of hazard analysis techniques. Some of the more commonly used techniques include Preliminary Hazard Analysis (PHA), Failure Modes and Effects Analysis (FMEA), Hazard and Operability Study (HAZOP), Fault Tree Analysis (FTA), and Event Tree Analysis (ETA).

Preliminary Hazard Analysis (PHA)

Preliminary Hazard Analysis or PHA is an analysis technique used in the early conceptual stages of design and development. The PHA is frequently used early in the conceptual stages prior to design completion. Typically, a team is used to identify potential hazards of the main system and possibly some of the major subsystems. It is used when there is limited information. Therefore, it is a high-level analysis and is not considered final. The PHA output can include ranking of hazards, operational constraints, recommended actions to eliminate or control the hazards, and perhaps additional safety requirements.

This technique is not discrete. It can use information including the results of the preliminary hazard list (PHL), lessons learned, system and component design data, safety design data, and malfunction data to identify potential hazard areas. PHA does not designate a specific technique; however, checklists and forms are commonly used.

[7] System safety analysis handbook, second edition, Prepared by the New Mexico Chapter of the System Safety Society, 1999. Available at http://www.system-safety.org/.

Failure Modes and Effects Analysis (FMEA)

Failure Modes and Effects Analysis or FMEA identifies failures of components, subsystems, and their effects on the system. In essence, “bottom up” approaches start with the system’s components. The standard, Procedures for performing a failure mode effects and criticality analysis [U.S. Department of Defense[8], 1980] provides detailed information on FMEA. Appendix C contains a sample form for FMEA.

This is a systematic technique to identify and analyze safety critical components and subsystems of a system. FMEA is most effectively conducted during the design phase thus enabling system design modifications to eliminate critical components or subsystems. Generally, the tabular format or spreadsheet is used. Usually, the analysis is conducted by a few engineers having a detailed understanding of the system and of the various failure modes for each component and subsystem.

Hazard and Operability Study (HAZOP)

Hazard and Operability Study or HAZOP is systematic and structured qualitative method of study conducted by a multi-disciplinary team. Guidewords are applied to various parameters to stimulate thinking concerning possible deviations. Because of these deviations, potential hazards, and causes are identified. Multiple references provide detailed information about HAZOP [Ministry of Defense 1998; Redmill et al[9]. 1999].

HAZOP had its beginnings in the chemical process industry where guidewords were designated for process industry parameters such as flow and pressure. HAZOP can be applied to a system, subsystem, process, or procedure and hardware and software. HAZOP can be easier to implement at the later stages when designs are firm rather than at conceptual phases. Thus, it is also well suited for hazard identification and analysis of modifications during the management of change (MOC) process.

[8] U.S. Department of Defense [1980]. Procedures for performing a failure mode effects and criticality analysis. Military Standard MIL-STD1629A.
[9] Redmill, F., Chudleigh, M., Catmur, J. [1999]. System safety: HAZOP and software HAZOP

HAZOP has been extended for the hardware and software of programmable systems [Redmill et al[10]. 1999]. The HAZOP is applied throughout the safety life cycle. Early in the life cycle, HAZOP is applied to top level design diagrams; as the design progresses, HAZOP is applied to more detailed electronics and software design diagrams.

Fault Tree Analysis (FTA)

Fault Tree Analysis or FTA is a logical “top down” method of structuring events and failures leading to a hazard. The Fault Tree Handbook [Vesely et al.[11], 1981] provides detailed information on FTA. FTA is a logical method of deduction utilizing a graphical depiction of events, faults, or logical combinations (Boolean expressions such as and, or, etc.) thereof. It begins at the top of the fault tree with an undesirable event. Next, the possible events and logical combinations are developed for the fault tree until the root causes are determined. The root causes can be triggering events or basic faults. It is best to use fault trees on the major events because the trees can grow quite large. FTA can be applied to hardware and to operational modes of the PPE. Fault trees are suited to analysis of static situations, thus dynamic situations involving timing are difficult to implement. In addition, fault trees can be qualitative or quantitative. A quantitative fault tree uses probabilities for the events and faults. Finally, the traditional fault tree for the system hardware has been extended to software fault tree analysis (SFTA).

[10] Redmill, F., Chudleigh, M., Catmur, J. [1999]. System safety: HAZOP and software HAZOP
[11] Vesely WE, Goldberg FF, Roberts NH, Haasl DF [1981]. Fault tree handbook (NUREG-0492). Washington, DC: U.S. Nuclear Regulatory Commission, Office of Nuclear Regulatory Research, Systems and Reliability Research.

Event Tree Analysis (ETA)

Event Tree Analysis or ETA is a logical, bottom-up graphical technique to determine outcomes from a single initiating hazardous event. The event tree is useful for determining the probability of each unwanted outcome resulting from a single initiating event. From this, one can determine which outcomes are the most severe or occur with the greatest frequency.

Potential or Predictive Human Error Analysis (HEA)

Potential or Predictive Human Error Analysis (HEA) is a team-based method similar in concept to HAZOP; however, this analysis focuses on human tasks and the associated error potential [AIHce 1994]. Human error causes fall into the following basic categories:

- Complexity: increases the likelihood of error
- Stress: increases the likelihood of error
- Fatigue: increases the likelihood of error
- Environment: adverse environments increase the likelihood of error
- Training: better training reduces the likelihood of error

The members of the team conducting the analysis consider these error causes as they conduct the analysis. A worksheet can be used to document the results.

Operating and Support Analysis (O&SA)

Operating and Support Analysis or O&SA seeks to identify hazards during operation and maintenance, find the root causes, determine the acceptable level of risk, and recommend risk reductions [Harms-Ringdahl[12] 1993]. An understanding of the operations, environment, and support (maintenance) philosophy (i.e. training, implementation, etc.) regarding the use of PPE needs to be analyzed. The O&SA is

[12] Harms-Ringdahl L.[1993]. Safety analysis - Principals and practices in occupational safety

used to identify hazards that may occur. The O&SA method is conducted by a team familiar with the system's operation and interaction with humans. Some of the items to be considered include:

- Operating during normal and abnormal conditions
- Making changes to the PPE.
- Maintaining the PPE electronics and software.
- Testing of the PPE
- Training personnel on the use and maintenance of the PPE
- Providing adequate documentation for the PPE.

Action Error Analysis (AEA)

Action Error Analysis or AEA is used to identify operator errors and the subsequent consequences. AEA specifically focuses on the interactions between humans and the system during operation, maintenance, and testing [Harms-Ringdahl L.1993]. The basic procedure is outlined as follows for operation and maintenance tasks:

- Identify operator tasks
- Detail the sub-tasks and actions for each task
- For each action, identify potential operator errors and consequences. As a guide, the following error types are considered for each action:
  - Error of omission (action not taken)
  - Wrong sequence of actions
  - Temporal errors (actions taken late or early)
  - Incorrect actions taken
  - Actions applied to the wrong interface object

The technique is well suited for automated or semi-automated processes with operator interfaces.

Interface Analysis

Interface analysis is used to identify hazards resulting from physical, functional, logical, and time-based interface inconsistencies. Interface analysis is applicable to all systems and interfaces. Numerous interfaces exist such as human-machine, hardware-software, hardware-hardware, and system-environment. The types of interface incompatibilities include the following:

- Human-Machine (i.e. usability)
- Environmental (i.e., temperature, moisture, dust, and vibration)
- Electrical (i.e., EMI, power sources, supply voltages, and data signals)
- Physical (i.e., rate and range of movement)
- Logical (i.e., conditional responses based on Boolean expressions)
- Temporal (i.e., clock times, response times, and delay times)

Inconsistencies can exist between adjacent, interconnected, interdependent, or interacting PPE components.

Sequentially Timed Events Plot (STEP)

Sequentially-Timed Events Plot or STEP is an event-driven approach to define systems, analyze operation, and to investigate mishaps. STEP is an analytical approach that graphically depicts sequentially timed events. Events are defined with formatted “building blocks” comprised of an “actor and action.” The event blocks are sequentially linked to graphically depict the flow of events that produce an outcome. The graphical depiction is useful for analyzing and defining events for a given system. STEP analysis can help discover and analyze problems; the analysis is also useful for assessing mitigation options. STEP is also used to analyze the types and sequences of events that lead to an incident.

APPENDIX B - RISK ANALYSIS

A risk analysis is used to determine which hazards have unacceptable risks and which hazards have acceptable risks and fall in the ALARP region. Hazards that fall in the ALARP region may be assigned a risk reduction category. The risk category specifies the safety design and performance requirements to reduce risk associated with the hazard to an acceptable level.

Risk categories may be defined using qualitative or quantitative methods. Qualitative techniques are applicable when it is not feasible to quantify risk. Common qualitative techniques include the risk assessment matrix, hazardous event severity matrix, and the risk graph. These techniques vary in terms of the type and detail of available information.

Layer of Protection Analysis (LOPA) [13] is a semi-quantitative technique of risk analysis used when multiple protection layers serve to reduce a risk to an acceptable level. Typically, the consequences are determined qualitatively and the frequency or likelihood is determined quantitatively. LOPA is helpful to determine if a safety function is adequately provided. A safety function can be implemented by a single protection layer consisting of a single device or safety system. Alternatively, it can be implemented using multiple protection layers. The number and type of protection layers depend on the design implemented to provide the safety function.

LOPA provides a more objective approach compared to purely qualitative techniques because the frequency or likelihood is determined quantitatively. LOPA enables more diversity and flexibility in the design. This can enable a design to potentially incorporate and account for existing protection layers as long as they are independent.

The **risk assessment matrix** [U.S. Department of Defense 1993] is the simplest. Risk is determined by using severity and frequency.

[13] First defined in *Part 1: Introduction to Functional Safety of this series) Layer of protection analysis (LOPA)*: An analysis that identifies risk reduction targets by evaluating selected risk scenarios.

The **hazardous event severity matrix** [IEC 1998f] is similar to the risk matrix, but it also takes independent layers of protection into account.

The **risk graph** uses severity and frequency, but it considers two additional parameters. The risk matrix is quite similar to the hazardous event severity matrix.

Severity categories are defined to provide a qualitative measure of the worst credible accident resulting from human error, environmental conditions, design inadequacies, procedural deficiencies, and system, subsystem or component failure.

A qualitative frequency may be derived from experience and evaluation of historical safety data from similar systems. Supporting rationale for assigning a mishap probability should be documented in hazard analysis reports.

The safety standards EN 954-1 [BSI 1997] and IEC 61508-5 [IEC 1998e] depict variations of risk graphs, yet they both use the basic concepts of qualitative estimates for risk in terms of severity, exposure, and avoidance.

APPENDIX C - ADDRESSING PPE ELECTRONIC COMPONENT FAULTS

Table C-1 is based on of IEC 61508 and ANSI/UL 1998. Definitions of these techniques can be found in Annex A of IEC 61508, Part 7 [IEC[14] 1998g] and in Section A7 of Appendix A of the second edition of UL 1998 [Underwriter Laboratories, Inc. 1998[15]]. It is being considered for update by the ANSI/UL 1998 Standards Technical Panel.

Table C-1.—PPE electronic component faults

COMPONENT	TECHNIQUE
Electronic	On-line monitoring
	Comparator
	Majority voter
	Tests by redundant hardware
	Dynamic principles
	Standard test access port and boundary-scan architecture
	Fail-safe hardware
	Monitored redundancy
	Hardware with automatic check
	Analog signal monitoring
Processing Units	Comparator
	Majority voter
	Self-test by software: limited number of patterns(one channel)
	Self-test by software: walking bit (one channel)
	Self-test supported by hardware (one-channel)
Processing Units (Continued)	Coded processing (one-channel)
	Reciprocal comparison by software

[14] IEC [1998g]. Functional safety of electrical/electronic/programmable electronic safety related systems. Geneva, Switzerland: International Electrotechnical Commission, IEC65108-7 Part 7: Overview of techniques and measures, version 4, May 12, 1998.
[15] Underwriters Laboratories, Inc. [http://www.ul.com/]. Date accessed: September 28, 2004.

Invariable memory ranges	Word saving multi-bit redundancy Modified checksum Signature of one word (8 Bit) Signature of a double word (16 Bit) Block replication
Variable memory ranges	RAM test "checkerboard" or "march" RAM test "walk-path" RAM test "Galpat" or "transparent Galpat" RAM test "Abraham" Parity-bit for RAM RAM monitoring with a modified hamming code Double RAM with hardware or software comparison and read/write test
Variable memory ranges	Failure detection by on-line monitoring Test pattern Code protection Multi-channelled parallel output Monitored outputs Input comparison/voting (1oo2, 2oo3 or better redundancy)
Data paths (internal communication)	One-bit hardware redundancy Multi-bit hardware redundancy Complete hardware redundancy Inspection using test patterns Transmission redundancy Information redundancy
Power supply	Over-voltage protection with safety shut-off or switch-over to second power unit

(Continued)

Table C 1 (Continued). PPE electronic component faults

COMPONENT	TECHNIQUE
Power supply (Continued)	Voltage control (secondary) with safety shut-off or switch-over to second power unit

	Power-down with safety shut-off or switch-over to second power unit Idle current principle
Program Sequencer/Instruction Counter	Watchdog with separate time basis without time-window Watchdog with separate time basis and time-window Logical monitoring of program sequence Combination of temporal and logical monitoring of programme sequences Temporal monitoring with on-line check
Clock	Watchdog with separate time base without time-window Watchdog with separate time base and time-window Logical monitoring of program sequence Temporal and logical monitoring Temporal monitoring with on-line check
Communication and mass-storage	Information exchange between E/E/PE safety-related system and process Information exchange between E/E/PE safety-related systems Separation of electrical energy lines from information lines Spatial separation of multiple lines Increase of interference immunity Anti-valent signal transmission
Sensors	Failure detection by on-line monitoring Idle current principle Analog signal monitoring Test pattern

(Continued)

Table C 1 (Continued). PPE electronic component faults

COMPONENT	**TECHNIQUE**
Sensors (Continued)	Input comparison/voting (1oo2, 2oo3 or better redundancy) Reference sensor

	Positive-activated switch
Final elements	Failure detection by on-line monitoring
	Monitoring of relay contacts
	Idle current principle
	Test pattern
	Monitoring
	Cross-monitoring of multiple actuators

APPENDIX D - DESCRIPTION OF SOFTWARE ANALYSIS AND TESTING METHODS

Analysis and test of software code starts at the module level, continues when software modules are integrated, when software modules are embedded in the processing hardware, when processing hardware is connected to sensing and controlling devices, and after deployment when configuration and other changes are made. Thus, software is tested at many layers typically starting with the smallest functional unit.

Methods for analyzing and testing software logic include both static analysis and dynamic testing. Static analysis methods are methods, which do not involve execution of the code. Dynamic methods are methods for evaluating software code by executing it. There are two types of dynamic methods: implementation-based and specification-based. Implementation-based testing is also called white-box or structural testing; specification-based testing is also called black box or functional testing. The appendix provides descriptions of some of the frequently used software analysis and test methods. It is not intended as a complete compendium of software analysis and testing methods.

While general guidelines exist for selecting and combining analysis and test methods and for deciding on which layer to apply which method, these guidelines are beyond the scope of this appendix.

STATIC ANALYSIS

Walkthroughs, Inspections, and Technical Reviews[16]

Walkthroughs are a process of reading or otherwise visually reviewing a document or code, especially comparing it with other system descriptions. The document or code is reviewed with respect to technique, style, possible errors,

[16] D.P. Freedman and G.M. Weinberg. Handbook of walkthroughs, inspections, and technical reviews: Evaluating programs, projects, and products, ISBN: 0932633196, Dorset House Publishing, New York, 2000.

violation of development standards, and other problems. Walkthroughs are characterized by the presence of the designer or programmer leading other members of the development team through the document or code while they ask questions and make comments. The process is guided by a checklist and governed by a set of detailed procedures.

Inspections are a review process similar to walkthroughs, except that the program author is not present. Work products are formally evaluated under the leadership of an independent moderator.

Technical Reviews are a review process similar to walkthroughs, with the exception that the customer is present**.**

Specifications/Requirements Reviews are walkthroughs or inspections conducted on a software requirements or specification document.

Design Reviews are walkthroughs or inspections conducted on a software design document to verify that requirements are correctly translated into design, that no additional functionality has been added, that interfaces are fully and correctly specified, and that design standards have been followed. A high-level design inspection validates the specification document against the requirements document. A low-level design inspection validates the design structure document against the requirements and design specification documents.

Code Reading/Reviews are walkthroughs or inspections conducted on software source code to verify that the design hierarchy (including interfaces) is correctly implemented and that coding standards have been followed. Checklists are used to identify common types of errors. Another approach is code reading, which is a process in which the code is read according to a systematic procedure with the goal of understanding how the code operates, and then of determining if it is correct with respect to its required functionality. Step-wise abstraction is a technique used for code reading in which higher levels of abstraction are derived from prime (or non-decomposable) subprograms for determining the actual functioning of the software and identifying errors or inconsistencies.

Mathematical Proofs of Correctness prove the correctness of a program without executing it, using theoretical and mathematical models and rules. The proof is accomplished by stating a number of pre and post conditions at various locations in the program. The proof consists of showing that the program transfers the pre-conditions into the post-conditions according to a set of logical rules, and that the program terminates.

Error and Anomaly Analysis is an analysis of the text of a software program with respect to syntax, data, and physical units to detect logic flaws and faults.

Structure Analysis involves evaluating control flow and code structures and subprogram calls to detect program faults. It is especially directed toward identifying and verifying sequences of invocations of functions.

IMPLEMENTATION-BASED TESTING

Implementation-Based Testing verifies that the software design elements have been implemented correctly. It examines the internal structure of a program (i.e., the code) to determine its elementary components (e.g., data structures, statements, decisions, paths) and constructs test cases that will execute those elementary structures or combinations of those elementary structures. The degree to which the test cases exercise the elementary structures of the code is the measure of completeness (or adequacy) of structural testing and is referred to as “test coverage.”

The various testing techniques are categorized according to coverage of a particular type of structure. This form of testing is frequently called white-box testing to emphasize that it is distinguished from specification or black box testing by its examination of the code. The specific structural techniques are differentiated by the type of structural element they address. The most commonly used techniques are statement, control flow, and data flow coverage. To illustrate differences in the various forms of structural testing, an example[17] is shown in Figure D-1. In this example we see that there are four statements; two decisions, each with two conditions and two outcomes; four possible paths through the code (a-b-d, a-b-e, a-c-d, and a-c-e); two computations; three data elements, one entry point, and one exit point.

```
M: Procedure (A,B,X):
  If A>1 AND B=0 then do;
                    X=X/A;
                      End;
     If A>2 or X>1 then do;
                    X=X+1;
                      End;
 End;
```

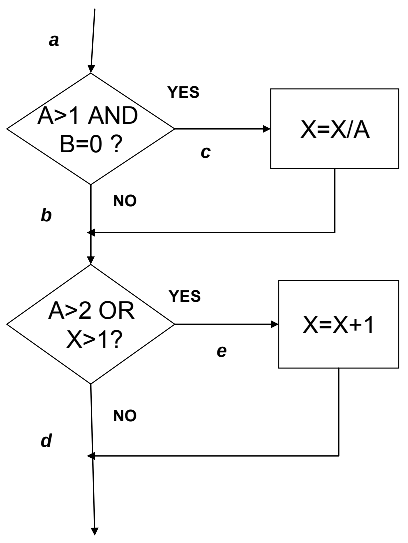

Figure D-1. Sample program code and control flow.

Statement Coverage requires that every source code statement be executed at least once. This is the easiest, but weakest form of structural testing. Referring to Figure D-1, we see that if a test case causes the Y branch to be taken at each decision point, then every statement will be executed. As indicated by Myers [1979], one test case (A = 2, B = '0, and X = '3) would suffice to test all statements; however, only one path (a-c-e) would have been exercised and the correctness of the decision statements would not have been addressed.

Control Flow Coverage is a form of testing in which the elementary components to be executed are chosen according to the flow of control from statement to

[17] Myers GJ [1979]. The art of software testing. New York, NY: John Wiley & Sons.

statement starting at the entry point and continuing to the exit point of the program. Thus, statements are selected for execution by each test case based on their relationship to the program's control flow. The number of control flow elements tested (versus the number of all control flow elements in the program) is the coverage metric associated with the test suite. Two control flow elements are commonly considered for control flow testing: decisions and paths. Decisions cause the control flow of a program to branch from one statement in a program to another, not necessarily its next linear successor.

The goal of **Decision Coverage** testing is to determine that the decisions are correctly implemented and to direct the control flow properly to result in the expected program output. For every decision, one or more conditions are evaluated, each of which has a set of possible outcomes and a set of possible outcomes of the decision, which result from the combination of the outcomes of its conditions. Branch coverage and multiple condition coverage are two of the several techniques that can be used to test decisions.

Branch Coverage attempts to invoke all outcomes of each decision and all entry points into the program. Branch coverage is also sometimes called decision coverage. Referring to Figure D-1, branch coverage would require both the Y and N branches to be taken at each decision point. As indicated by two test cases (A = 3, B = 0, X = 3 and A = 2, B = 1, X = 1) would provide 100% branch coverage; however, only two paths (a-c-d and a-b-e) would have been exercised and, just as in the statement coverage case, the correctness of the decision statements would not have been addressed.

Multiple Condition Coverage attempts to address the inadequacy of statement and branch coverage for testing the correctness of decision statements by ensuring that the conditions in the decision statements are completely exercised. It requires that all possible combinations of condition outcomes for each decision and all entry points into the program be invoked. This is sometimes extended to require that all combinations of input at each decision statement, and at any computation or assignment used as a flag for a decision, are executed. As detailed in the example in Figure D-1 there are eight combinations of conditions to be covered by test cases: (1) A > 1, B = 0; (2) A > 1, B ≠ 0; (3) A ≤ 1, B = 0; (4) A ≤ 1, B ≠ 0; (5) A = 2, X > 1; (6) A = 2, X ≤ 1; (7) A ≠ 2, X > 1; and (8) A ≠ 2, X ≤ 1. These combinations can be covered by four test cases: (1) A = 2, B = 0, X = 4; (2) A = 2, B=1, X = 1; (3) A = 1, B=0, X = 2; and (4) A = 1, B = 1, X = 1; however, the a-c-d path would not have been exercised.

Thus, while decision coverage is more thorough than statement coverage, it still does not provide a high level of coverage of all the elementary components, in part because of its failure to exercise all combinations of branches.

Path Coverage testing addresses this area by identifying and exercising complete execution paths through the program. A path is a subset of program statements that are executed in sequence according to the program's control flow, where the first statement in the path is the first statement to be executed after the entry point into the program and the last statement in the path is the last statement to be executed before the exit point from the program. A major difficulty encountered in path testing is that even small programs can have very large numbers of unique paths, large enough, in fact, to make exhaustive path testing infeasible. Even if all paths could be tested, all defects would not necessarily be detected because path testing cannot detect missing paths or data-sensitivity errors. Two techniques for selecting the appropriate paths are Decision-to-Decision Path Testing and Linear Code Sequence and Jump (LCSAJ) Measures.

Decision-to-Decision Path testing exercises paths consisting of all program statements occurring between one decision statement and the next sequential decision statement and that are made executable by the selection of the outcome of the initial decision statement. These paths, also called DD paths, are identified from a directed graph of the program consisting of nodes identifying program code statement segments and directed arcs identifying control transfers from one node to another. If each DD path is executed, then each statement is executed at least once and all possible outcomes of each decision are exercised[18].

Linear Code Sequence and Jump (LCSAJ) Testing is a technique that executes a sequence of consecutive code statements bounded by either the entry point into the code or a jump (branch) from one code location to another and by either the end of the program or the beginning of another jump. The code segments that form an LCSAJ are identified from the text of the program, not from a directed graph. LCSAJs are combined to form sub paths. A series of LCSAJ measures can be defined based on the number of LCSAJs exercised versus the total number of LCSAJs and the number n of LCSAJs constituting a sub path versus the total number of sub paths containing n or fewer LCSAJs.

Data Flow Coverage is a form of testing in which the elementary components to be executed are chosen according to the flow of data from statement to statement starting at the entry point and continuing to the exit point of the program. Thus, statements are selected for execution by each test case based on their relationship to the program's data elements, i.e., testing is based on the existence of a set of paths between definitions and assignments to data elements and references to and uses of those data elements. The goal of data flow coverage testing is to uncover data-sensitive errors and thus addresses the deficiencies in control flow testing in that area. There are several coverage criteria in this class, including the following: (1) all uses of a variable in any computation must be exercised (designated all-c-uses), (2) all uses of a variable in a branch must be exercised and in such a way as to cause all outcomes of the branch to be exercised (designated all-p-uses), (3) all-c-uses and all-p-uses (designated all-uses), (4) every assignment to a variable must be exercised and every loop less path between the assignment and the use of the variable must be traversed (designated all-du-paths), (5) each variable input to a subroutine is examined after the execution of the subroutine to demonstrate that there was no

[18] Miller E.F. Jr., Paige MR, Benson JP, Wisehart W.R. [1981]. Structural techniques of program validation. Tutorial: software testing and validation techniques, second edition. Los Alamitos, Calif.: IEEE Computer Society Press, pp. 304-307.

unexpected change in its value.

SPECIFICATION-BASED TESTING

Specification-Based Testing safeguards that all requirements are met; tests that intended functions are performed fully and correctly; and exercises all paths that are specified for performance of a required process. Software is treated as a black box, i.e., test cases are designed and input selected based on requirements and specifications without regard to the actual code, the software is executed using those inputs, and the resulting outputs are compared to the outputs that were expected based on the specifications and requirements. Complete testing would require that all possible values of all variables defined by the requirements and specification be tested. Since this is impossible, it is necessary to have a way of selecting the best subset of all possible inputs. Frequently used selection techniques are equivalence partitions, boundary values, special values, and statistical usage testing.

Equivalence Partitions is a technique in which the input and output ranges are partitioned into at least two equivalence classes and input values are selected from each class. Any input value selected from a particular equivalence class should result in an output equivalent to that resulting from any other input value selected from that class. Both invalid and valid classes must be considered.

Boundary Values is a technique in which the set of inputs is selected from the boundary values of the equivalence classes that partition the input ranges defined by the specifications. The boundary values of an equivalence class are values outside the class that bound the values in the class. The minimum and maximum values in the class are also included in the boundary values.

Interface Testing detects errors in the interfaces between software programs. Interface testing is particularly important if the interfaces do not contain assertions that detect incorrect parameter values. Interface testing is also important when pre-existing software programs are modified.

Special Values is a technique in which the set of inputs is selected based on values that are likely to be associated with faults. These values may be selected based on the hazard analysis methods conducted, such as FTA and FMEA, or based on performance requirements. Some specific types of special values are distinct values and zero values. Distinct values are assigned to input variables for some particular purpose, such as to ensure that the values of elements of an array differ and that particular indices are different in particular arrays. Zero values (zeroes) are assigned to variables in arithmetic expressions, and input variables are selected to cause internal variables to take on the value 0.

Usage Testing (statistical/random or use case testing) is a technique in which the set of inputs are a selected sample of all possible input values. The sample is selected based on randomness, typical user profiles or use cases.